Available online at www.sciencedirect.com
ScienceDirect
Theoretical Computer Science 386 (2007) 38–56
Theoretical Computer Science
www.elsevier.com/locate/tcs

Automata theory based on quantum logic: Reversibilities and pushdown automata☆

Daowen Qiu*

Department of Computer Science, Zhongshan University, Guangzhou 510275, PR China

Received 21 April 2004; received in revised form 29 March 2007; accepted 16 May 2007

Communicated by G. Brassard

Abstract

Automata theory based on quantum logic, called *l-valued finite automata* (*l*-VFAs), may be viewed as a logical approach to quantum computing. This work is mainly divided into two parts: one part deals with *reversibility* of *l*-VFAs, and the other establishes a basic framework of *l-valued pushdown automata* (*l*-VPDAs). First we provide some preliminaries concerning quantum logic and *l*-VFAs, and we prove a useful property of *l*-valued successor and source operators. Then we clarify the relationships between various reversibilities closely related to quantum finite automata in the literature. In particular, we define a reversibility of *l*-VFAs which is termed as *retrievability*, and we clarify the relationships between a number of different fashions regarding retrievability of *l*-VFAs. We prove that some of them are equivalent, but for the others to be equivalent the truth-value set is required to satisfy a certain condition. This is an essential difference from the classical situation. Afterwards, we introduce *l*-VPDAs and show that the class of the languages accepted by *l*-VPDAs *by empty stack* coincides with that accepted by *l*-VPDAs *by final state*. Finally, we provide some examples of *l*-VFAs and conclude with some remarks.

Keywords: Quantum finite automata; Quantum logic; Orthomodular lattices; Finite automata; Pushdown automata; Quantum computation

1. Introduction

Over the past two decades, quantum computing has attracted extensive attention in the academic community [24, 30]. Let us briefly recall some pioneer works in this area. In 1980, Benioff [7] first considered that computing devices in terms of the principles of quantum mechanics could be at least as powerful as classical computers. Then Feynman [20] pointed out that there appears to be no efficient way of simulating a quantum mechanical system on a classical computer, and he suggested that a computer based on quantum physical rules might be able to carry out the simulation efficiently. By formalizing their ideas, Deutsch [17] in 1985 re-examined the Church–Turing Principle and defined quantum Turing machines.

☆ This research is supported by the National Natural Science Foundation (Nos. 90303024, 60573006), and the Research Foundation for the Doctoral Programme of Higher School of Ministry of Education (No. 20050558015) of China.

* Corresponding author. Tel.: +86 02038458089; fax: +86 02084037549.
E-mail address: issqdw@mail.sysu.edu.cn.

0304-3975/$ - see front matter
doi:10.1016/j.tcs.2007.05.026

Quantum computation from the viewpoint of complexity theory was studied first by Bernstein and Vazirani [14], and they described an efficient universal quantum Turing machine that can simulate a large class of quantum Turing machines. Yao [44] demonstrated the equivalence between quantum Turing machines and quantum circuits. The most remarkable discoveries are Shor's quantum polynomial time algorithm for factorizing integers [43] and Grover's quantum searching algorithm [22]. Since the computational complexity of integer factoring is the basis of the RSA public cryptosystem, Shor's finding underlines the power of quantum computers. Actually, since then, the research regarding quantum computing has been more and more active [24,30].

Quantum finite automata [29,27] can be thought of as theoretical models of quantum computers with finite memory, as finite automata are in the theory of computation [25]. To date, finite automata have been significantly developed. In the 1960's, Rabin [38], Salomaa [40], Ellis [19], Santos [41], etc. formulated probabilistic automata and probabilistic grammars. In a probabilistic automaton [38], each transition is equipped with a number in the unit interval which indicates the probability of the occurrence of the transition; by contrast, in a quantum finite automaton [29,27] we associate each transition with a vector in a Hilbert space which is interpreted as the probability amplitude of the transition, the transition function complies with the laws of quantum mechanics and therefore is required to be unitary (reversible), and the probability of computing an input results from a certain measurement. As for automata theory based on quantum logic—*l-valued finite automata* (*l*-VFAs) [45,46,34,36], each transition is assigned an element (that can be viewed as a closed subspace or a projector of a Hilbert space) in an orthomudular lattice.

Moore and Crutchfield [29], and Kondacs and Watrous [27] initiated the study of quantum finite automata, and they gave different definitions of quantum finite automata in terms of the measurement times. In [29], one measurement is performed at the end of each computation, and such models are usually called *measure-once automata* [24], while in [27] measurement is implemented after each evolution of states, and quantum finite automata defined in this way are called *measure-many automata*. Another kind of important computing models—quantum pushdown automata, was considered in [21,37]. To date, quantum finite automata have been studied extensively by many authors (for example, see [2–4,9,13,21,23,31,35,37] and references therein). One of the main concerns with these computing models is to clarify their power as compared with classical models. Indeed, Brodsky and Pippenger [13] showed that measure-once automata accept group languages with bounded-error probability, a proper subset of regular languages; Kondacs and Watrous [27] demonstrated that two-way quantum finite automata are more powerful than their classical counterparts; we [35] verified an intrinsic difference between quantum sequential machines and stochastic sequential machines, which answers partially a question asked by Gudder [23]. To a great extent, these findings are closely related to the unitarity and superposition property of quantum physics. If the entanglement property of quantum mechanics is employed into quantum finite automata, we may expect to discover some further essential characteristics.

A more basic issue regarding the models of quantum computation is *l*-VFAs, considered first by Ying [45,46] and then studied by us [34,36], as well as investigated by the others (e.g., [28,16]). Quantum logic was introduced by Birkhoff and von Neumann [12] in 1936 as the logic of quantum mechanics, and it stemmed from von Neumann's Hilbert space formalization of quantum mechanics in which the behaviour of a quantum mechanical system is described by a closed space of a Hilbert space. By noting that the set of all closed subspaces of a Hilbert space consists of an orthomodular lattice, it was suggested that orthomodular lattices would be used as the algebraic version of the logic of quantum mechanics, just like Boolean algebra acting as an algebraic counterpart of classical logic. Usually, orthomodular lattices are thought of as quantum logic [33]. In *l*-VFAs, the truth-value set of quantum logic is understood as an orthomodular lattice, and each transition of an *l*-VFA is assigned an element in the orthomodular lattice. Also, Ying [46] proved that the product operation of *l*-VFAs holds if and only if the truth-value lattice satisfies the usual distributivity.

We [36] further elaborated on *l*-VFAs. In particular, we found more intrinsic connections between the properties of *l*-VFAs and the properties of its truth-value set of logic. To be more precise, we showed that some properties of *l*-VFAs hold if and only if the truth-value set of logic satisfies the distributivity of $\vee$ over $\wedge$, i.e. the orthomodular lattice underlying logic reduces to a Boolean algebra, while some properties in *l*-VFAs hold if and only if the truth-value set of logic satisfies the distributivity of $\vee$ over &, which is strictly weaker than the distributivity of $\vee$ over $\wedge$.

This paper is a continuation of [36]. The main contributions can be summarized as follows: first we deal with the reversibility of *l*-VFAs by proving a number of equivalent characterizations; then we establish the basic setting of *l-valued pushdown automata* (*l*-VPDAs). As is well known, energy is an important computational resource. By virtue of Landauer's principle, energy consumption in computation is deeply linked to the reversibility of the computation; conversely, in reversible computation, it is necessary that no energy is dissipated. In 1973, Bennett [8] showed that

any classical Turing machine can be efficiently simulated by a reversible Turing machine. Quantum computing is unitary and thus reversible. In this regard, studying the reversibility of computation is of great importance in quantum computing. Brodsky and Pippenger [13] as well as Ambainis and Freivalds [2] studied the power of reversible finite automata for recognizing languages by comparing them with quantum finite automata, in which a reversible finite automaton is indeed defined as a quantum finite automaton whose transition amplitudes are either 0 or 1.

In Section 2 we provide some preliminaries concerning orthomodular lattice-valued logic and l-VFAs, and we prove a property of l-valued successor and source operators that will be used later on. Then, in Section 3 we deal with the relationships between a number of different definitions of reversibility in l-VFAs. We first clarify the relations among various reversibilities related to quantum finite automata, and then we define a reversibility of l-VFAs which is termed as *retrievability* to avoid confusion with the reversibilities in the literature [13,2]. (Notably, retrievability of classical finite automata was first defined by Bavel and Muller [11,5].) In particular, we clarify the relationships between a number of retrievabilities of l-VFAs and demonstrate that, some of them are equivalent, but for the others, they are equivalent if and only if the truth-value set satisfies distributivity. In contrast, all these definitions concerning retrievability are exactly equivalent in classical finite automata [11]. This is also an essential difference between l-VFAs and classical finite automata.

Pushdown automata are another kind of important models of computation after finite automata [25]. In Section 4, we introduce *l-valued pushdown automata* (l-VPDAs), a generalization of classical pushdown automata. In particular, we show that the class of the languages accepted by l-VPDAs *by empty stack* coincides with that accepted by l-VPDAs *by final state*.

In Section 5 we state some existing applications of quantum logics; in particular, we present some examples of l-VFAs, and these examples imply that, though some results do *not* hold in *all* l-VFAs, they may hold in *some* l-VFAs. To conclude, in Section 6 some remarks are included and a number of related issues are raised.

In general, notation used in this paper will be explained whenever new symbols appear.

2. Preliminaries

2.1. Quantum logic: Complete orthomodular lattice-valued logic

In the interest of readability, we briefly recall some notions and terminologies in quantum logic. For the details, we refer to [33,36,39,45,46]. A 7-tuple $l = \langle L, \leq, \wedge, \vee, \perp, 0, 1\rangle$ is called a complete orthomodular lattice, if it satisfies the following conditions **(1)** and **(2)**:

(1) $\langle L, \leq, \wedge, \vee, 0, 1\rangle$ is a complete lattice, 0 and 1 are the least and greatest elements of L, respectively, $\leq$ is the partial ordering in L, and for any $M \subseteq L$, $\wedge M$ and $\vee M$ stand for the greatest lower bound and the least upper bound of M, respectively.

(2) $\perp$ is a unary operation on L, called orthocomplement, and it is required to satisfy the following conditions: for any $a, b \in L$,

(2.1) $a \wedge a^{\perp} = 0, a \vee a^{\perp} = 1$.

(2.2) $a^{\perp\perp} = a$.

(2.3) $a \leq b$ implies $b^{\perp} \leq a^{\perp}$.

(2.4) $a \geq b$ implies $a \wedge (a^{\perp} \vee b) = b$.

A quantum logic is a complete orthomodular lattice-valued logic (called l-valued logic). In this paper, we use Sasaki arrow as the implication operator. Sasaki arrow is defined as follows: for any $a, b \in L$,

(3) $a \to b \stackrel{def}{=} a^{\perp} \vee (a \wedge b)$.

The conjunction in a quantum logic is usually interpreted as the meet operation of the truth-value lattice. Because the meet operation is not conjugate to Sasaki arrow, Román and Rumbos [39] introduced a new conjunction operator, namely, the multiplication &, which is defined as follows: for all $a, b \in L$,

(4) $a\&b \stackrel{def}{=} (a \vee b^{\perp}) \wedge b$.

Some of the properties of Sasaki arrow and the multiplication & that we will use later on are provided as follows:

(4.1) $a\&b \leq c$ iff $a \leq b \to c$.

(4.2) $a \leq b$ iff $a \to b = 1$.

(4.3) $(a \to b)\&a \leq b$.

(4.4) $a\&(a \to b) \le b$.
(4.5) $0 \to a = 1 = a \to 1$.
(4.6) $a\&b \le b$.
(4.7) $a \wedge b \le a\&b$.
(4.8) L is a Boolean algebra (i.e. the distributivity of $\vee$ over $\wedge$ holds) iff any one of the following holds:
(i) & is commutative, i.e. $a\&b = b\&a$ for any $a, b \in L$.
(ii) $b \le c \Rightarrow a\&b \le a\&c$ for any $a, b, c \in L$.
The bi-implication operator corresponding to Sasaki arrow is defined as follows: for all $a, b \in L$,
(5) $a \leftrightarrow b \stackrel{def}{=} (a \to b) \wedge (b \to a)$.

Let $l = \langle L, \le, \wedge, \vee, \perp, 0, 1\rangle$ be a complete orthomodular lattice, and let $\to$ be Sasaki arrow. The syntax of l-valued logic is similar to that of classical first-order logic. We have three primitive connectives $\neg$ (negation), $\wedge$ (conjunction) and $\to$ (implication), and a primitive quantifier $\forall$ (universal quantifier). The connectives $\vee$ (disjunction) and $\leftrightarrow$ (bi-implication) and the existential quantifier $\exists$ are defined in terms of $\neg$, $\wedge$, $\to$ and $\forall$ in the usual way.

In addition, we need to use some set-theoretical formulas. Let $\in$ (membership) be a binary (primitive) predicate symbol. Then $\subseteq$ (inclusion) and $\equiv$ (equality) can be defined with $\in$ as usual. The semantics of l-valued logic is given by interpreting the connectives $\neg$, $\wedge$ and $\to$ as the operations $\perp$, $\wedge$ and $\to$, on L, respectively, and interpreting the quantifier $\forall$ as the greatest lower bound in L. Moreover, the truth value of set-theoretical formula $x \in A$ is $\lceil x \in A\rceil = A(x)$. In this paper, the set A and its characteristic function are identified. In the l-valued logic, 1 is the unique designated truth value; a formula φ is valid iff its truth-value $\lceil\varphi\rceil$ is 1.

2.2. *l-valued finite automata*

A finite automaton whose initial and final states are omitted is defined as a triple (Q, Σ, δ), where Q denotes the finite set of states, Σ the finite input alphabet, and δ a transition relation, i.e. a partial function from $Q \times \Sigma$ to Q. In terms of this definition, l-VFAs are defined as follows:

Definition 2.1 (*[36]*). An l-VFA is defined as a triple $\mathcal{M} = (Q, \Sigma, \delta)$, where Q denotes a finite set of states, Σ a finite input alphabet, and δ the transition relation, i.e. a mapping from $Q \times \Sigma \times Q$ to L. In other words, δ is an l–valued subset of $Q \times \Sigma \times Q$. (In this paper, the set of all l-valued subsets of set X is denoted by L^X.)

By the way, l-VFAs defined above are instead called *l-valued automata* in [36]. In view of the finiteness of states, in this paper we call it l-VFAs.

To describe the transitions enabled by a string of input symbols, δ may be naturally extended to $\delta^* : Q \times \Sigma^* \times Q \to L$ as follows: for any $p, q \in Q$, if $q = p$, then $\delta^*(p, \epsilon, q) = 1$; otherwise, $\delta^*(p, \epsilon, q) = 0$, and

$$\delta^*(p, xa, q) = \bigvee\{\delta^*(p, x, r) \wedge \delta^*(r, a, q) : r \in Q\}$$

for any $x \in \Sigma^*$ and $a \in \Sigma$, where $\Sigma^* = \bigcup_{k=0}^{\infty} \Sigma^k$, $\Sigma^{(0)} = \{\epsilon\}$, and $\Sigma^k = \{\sigma_1\sigma_2 \cdots \sigma_k : \sigma_i \in \Sigma, i = 1, 2, \ldots, k\}$, where ϵ represents an empty string in this paper.

Definition 2.2 (*[36]*). Let $\mathcal{M} = (Q, \Sigma, \delta)$ be an l-VFA. Then the *successor* and *source* operators S and R from L^Q to L^Q are respectively defined as follows: for any $A \in L^Q$ and any $q, p \in Q$,

$$S(A)(q) \stackrel{def}{=} \bigvee\{A(p) \wedge \delta^*(p, x, q) : p \in Q, x \in \Sigma^*\},$$

$$R(A)(p) \stackrel{def}{=} \bigvee\{A(q) \wedge \delta^*(p, y, q) : q \in Q, y \in \Sigma^*\}.$$

From the above definitions it directly follows that for any $p \in Q$,

$$S(A)(p) \ge A(p) \quad \text{and} \quad R(A)(p) \ge A(p).$$

In order to explain the definition of *l-valued subautomata* more clearly, we first recall the definition of subautomata in classical automata theory [5]. A subautomaton of a finite automaton means a subset of the set of states satisfying the closure property under the successor operator. More specifically, let $M = (Q, \Sigma, \delta)$ be a classical automaton (usually, we use M and $\mathcal{M}$ to denote a classical automaton and an l-VFA, respectively), and $A \subseteq Q$. We call (A, Σ, δ) a

subautomaton of M, if δ is also a partial function from $A \times \Sigma$ to A, namely, for any state $q \in A$, and any $\sigma \in \Sigma$, it satisfies either $\delta(q, \sigma) \in A$ or $\delta(q, \sigma)$ undefined. In l-VFA $\mathcal{M} = (Q, \Sigma, \delta)$, as a generalization, a subset of Q is naturally thought of as an l-valued subset of Q. Clearly, any classical subset of Q can be identified with a special l-valued subset of Q, since any $A \subseteq Q$ means that $A(q) = 1$ if $q \in A$; and $A(q) = 0$, otherwise, where 1 and 0, as above, are the greatest and least elements of L, respectively. According to the definition of classical subautomata, we formally define l-valued subautomata as follows:

Definition 2.3 (*[36]*). Let $\mathcal{M} = (Q, \Sigma, \delta)$ be an l-VFA. Then for any $A \in L^Q$, we call A an *l-valued subautomaton* of $\mathcal{M}$ if

$$\overset{l}{\models} (\forall q \in Q)(q \in A \to (\forall p \in Q)(\forall x \in \Sigma^*)((q, x, p) \in \delta^* \to p \in A)),$$

equivalently, for any $q \in Q$,

$$A(q) \leq \bigwedge\{\delta^*(q, x, p) \to A(p) : x \in \Sigma^*, p \in Q\}.$$

We show that the above definition regarding l-valued subautomata conforms to the classical case. On the one hand, let A be a classical subset of Q, and A is an l-valued subautomaton of $\mathcal{M}$. Then for any $q \in A$, by means of the above inequality in Definition 2.3, we have

$$\begin{aligned} 1 = A(q) &\leq \bigwedge\{\delta^*(q, x, p) \to A(p) : x \in \Sigma^*, p \in Q\} \\ &= \bigwedge\{\delta(q, \sigma, p) \to A(p) : \sigma \in \Sigma, p \in A\} \wedge \bigwedge\{\delta(q, \sigma, p) \to A(p) : \sigma \in \Sigma, p \notin A\} \\ &= \bigwedge\{\delta(q, \sigma, p) \to 1 : \sigma \in \Sigma, p \in A\} \wedge \bigwedge\{\delta(q, \sigma, p) \to 0 : \sigma \in \Sigma, p \notin A\} \\ &= \bigwedge\{\delta(q, \sigma, p)^\perp \vee \delta(q, \sigma, p) : \sigma \in \Sigma, p \in A\} \wedge \bigwedge\{\delta(q, \sigma, p)^\perp : \sigma \in \Sigma, p \notin A\} \\ &= \bigwedge\{\delta(q, \sigma, p)^\perp : \sigma \in \Sigma, p \notin A\}, \end{aligned}$$

from which it follows that if $p \notin A$, then it must be $\delta(q, \sigma, p) = 0$. In other words, for any $q \in A$ and any $p \notin A$, the degree to which the machine changes to state p from the current state q after reading σ, is zero. This accords exactly with the definition of classical subautomata. In particular, if $A = Q$, then for any $q, p \in A$, $A(q) = A(p) = 1$, and therefore, it clearly satisfies the condition of l-valued subautomata. As a result, l-VFA $\mathcal{M}$ itself is also an l-valued subautomaton.

On the other hand, if (A, Σ, δ) is a classical subautomaton of classical automaton $M = (Q, \Sigma, \delta)$, then for any $q \in A$, any $\sigma \in \Sigma$, and $p \notin A$, $\delta(q, \sigma, p) = 0$, i.e. $\delta(q, \sigma, p)^\perp = 1$. This also satisfies the logical implication condition of l-valued subautomata in Definition 2.3, since for any $q \in A$, any $\sigma \in \Sigma$, and any $p \notin A$, $A(q) = 1$, and by the preceding calculation,

$$\bigwedge\{\delta^*(q, x, p) \to A(p) : x \in \Sigma^*, p \in Q\} = \bigwedge\{\delta(q, \sigma, p)^\perp : \sigma \in \Sigma, p \notin A\} = 1.$$

Proposition 2.1 (*[36], Corollary 4.4*). *The following six statements are equivalent:*

(i) *L satisfies the distributive law: $a \wedge (b \vee c) = (a \wedge b) \vee (a \wedge c)$ for all $a, b, c \in L$, that is to say, L is a Boolean algebra.*

(ii) *For any $a, b \in L$, $b \wedge (b^\perp \vee a) \leq a$, i.e. $a \& b \leq a$.*

(iii) *For any $a, b \in L$, $b^\perp \vee (b \wedge a) \geq a$.*

(iv) *For any l-VFA $\mathcal{M} = (Q, \Sigma, \delta)$ and for any $A \in L^Q$, if $\overset{l}{\models} S(A) \equiv A$, then $\overset{l}{\models} R(A^\perp) \equiv A^\perp$.*

(v) *For any l-VFA $\mathcal{M} = (Q, \Sigma, \delta)$ and for any $A \in L^Q$, if $\overset{l}{\models} R(A^\perp) \equiv A^\perp$, then $\overset{l}{\models} S(A) \equiv A$.*

(vi) *For any l-VFA $\mathcal{M} = (Q, \Sigma, \delta)$ and for any $A \in L^Q$, if $\overset{l}{\models} S(A) \equiv A$, then A is an l-valued subautomaton of $\mathcal{M}$.*

Proposition 2.2. *The following three statements are equivalent:*

(i) *L satisfies the distributive law: $a \wedge (b \vee c) = (a \wedge b) \vee (a \wedge c)$ for all $a, b, c \in L$, that is, L is a Boolean algebra.*

(ii) *For any l-VFA $\mathcal{M} = (Q, \Sigma, \delta)$, and for any $A \in L^Q$,*

$$\overset{l}{\models} S(S(A)) \equiv S(A).$$

(iii) *For any l-VFA* $\mathcal{M} = (Q, \Sigma, \delta)$*, and for any* $A \in L^Q$*,*

$$\overset{l}{\models} R(R(A)) \equiv R(A).$$

Proof. (i) $\Rightarrow$ (ii): Since L satisfies the distributivity, for any $p \in Q$, we have

$$\begin{aligned}
S(S(A))(p) &= \bigvee_{q\in Q}\bigvee_{x\in\Sigma^*}(\delta^*(q,x,p)\wedge S(A)(p))\\
&= \bigvee_{q\in Q}\bigvee_{x\in\Sigma^*}\left(\delta^*(q,x,p)\wedge\left(\bigvee_{r\in Q}\bigvee_{y\in\Sigma^*}(\delta^*(r,y,q)\wedge A(r))\right)\right)\\
&= \bigvee_{q\in Q}\bigvee_{x\in\Sigma^*}\bigvee_{r\in Q}\bigvee_{y\in\Sigma^*}(\delta^*(r,y,q)\wedge\delta^*(q,x,p)\wedge A(r))\\
&= \bigvee_{r\in Q}\bigvee_{y\in\Sigma^*}\bigvee_{x\in\Sigma^*}\left(\bigvee_{q\in Q}(\delta^*(r,y,q)\wedge\delta^*(q,x,p))\wedge A(r)\right)\\
&= \bigvee_{r\in Q}\bigvee_{y\in\Sigma^*}\bigvee_{x\in\Sigma^*}(\delta^*(r,yx,p)\wedge A(r))\\
&= \bigvee_{r\in Q}\bigvee_{z\in\Sigma^*}(\delta^*(r,z,p)\wedge A(r))\\
&= S(A)(p).
\end{aligned}$$

So $\overset{l}{\models} S(S(A)) \subseteq S(A)$ holds.

(ii) $\Rightarrow$ (i): Let $a, b, c \in L$. Take $Q = \{p, r_1, r_2, q\}$, $\Sigma = \{\sigma\}$, $\delta(p,\sigma,q) = a$, $\delta(r_1,\sigma,p) = b$, $\delta(r_2,\sigma,p) = c$, $\delta(r_1,\sigma,q) = \delta(r_2,\sigma,q) = a$, and $A \in L^Q$ with $A(r_1) = b$, $A(r_2) = c$. Then we have

$$\begin{aligned}
S(A)(q) &= \bigvee_{r\in Q}\bigvee_{x\in\Sigma^*}(\delta^*(r,x,q)\wedge A(r))\\
&= (a\wedge b)\vee(a\wedge c),
\end{aligned}$$

and $S(A)(p) = b \vee c$, $S(A)(r_1) = A(r_1) = b$, $S(A)(r_2) = A(r_2) = c$, as well as

$$\begin{aligned}
S(S(A))(q) &= \bigvee_{r\in Q}\bigvee_{x\in\Sigma^*}(\delta^*(r,x,q)\wedge S(A)(r))\\
&= \bigvee_{x\in\Sigma^*}(\delta^*(p,x,q)\wedge S(A)(p))\\
&\quad\vee\bigvee_{x\in\Sigma^*}(\delta^*(r_1,x,q)\wedge S(A)(r_1))\\
&\quad\vee\bigvee_{x\in\Sigma^*}(\delta^*(r_2,x,q)\wedge S(A)(r_2))\vee S(A)(q)\\
&= (a\wedge(b\vee c))\vee(a\wedge b)\vee(a\wedge c)\vee(a\wedge b)\vee(a\wedge c)\\
&= a\wedge(b\vee c)
\end{aligned}$$

since $a\wedge(b\vee c) \geq (a\wedge b)\vee(a\wedge c)$ always holds. Thus it follows from the statement (ii) that $a\wedge(b\vee c) = (a\wedge b)\vee(a\wedge c)$.

(i) $\Rightarrow$ (iii): Similar to the proof of (i) $\Rightarrow$ (ii).

(iii) $\Rightarrow$ (i): Let $a, b, c \in L$. Take $Q = \{p, r_1, r_2, q\}$, $\Sigma = \{\sigma\}$, $\delta(q,\sigma,p) = a$, $\delta(p,\sigma,r_1) = b$, $\delta(p,\sigma,r_2) = c$, $\delta(q,\sigma,r_1) = \delta(q,\sigma,r_2) = a$, and $A \in L^Q$ with $A(r_1) = b$, $A(r_2) = c$. Then, similar to the calculation in (ii) $\Rightarrow$ (i), one obtains that

$$\begin{aligned}
R(A)(q) &= \bigvee_{r\in Q}\bigvee_{x\in\Sigma^*}(\delta^*(q,x,r)\wedge A(r))\\
&= (a\wedge b)\vee(a\wedge c),
\end{aligned}$$

and $R(A)(p) = b \wedge c$, $R(A)(r_1) = A(r_1) = b$, $R(A)(r_2) = A(r_2) = c$ as well as $R(R(A))(q) = a \wedge (b \vee c)$. So, by the statement (iii), we have $a \wedge (b \vee c) = (a \wedge b) \vee (a \wedge c)$ and thus complete the proof. □

3. Reversibilities of l-valued finite automata

3.1. Reversibilities related to quantum finite automata

As mentioned above, reversibility is a pivotal concept in quantum computation because the unitarity of quantum computation requires the computation to be reversible. Reversibility of finite automata was early dealt with by many authors such as Bavel and Muller [11,5], Angluin [1], Pin [32], and the authors in References of [11,32]. However, when they defined reversibility of finite automata, the requirement of *deterministic finite automata* (DFAs) [25] was not considered; in other words, the transition functions in their reversible automata may be partial. Bavel and Muller [11] defined that a *reversible finite automaton* is a finite automaton $M = (Q, \Sigma, \delta)$ that for any input symbol $\sigma \in \Sigma$ there exists a string $x \in \Sigma^*$ such that for any state $q \in Q$, $\delta(q, \sigma x) = q$, while they defined a finite automaton $M = (Q, \Sigma, \delta)$ to be *retrievable* if for any state $q, p \in Q$, and any $\sigma \in \Sigma$, $\delta(q, \sigma) = p$ implies that there exists string $x \in \Sigma^*$ such that $\delta(p, x) = q$. Intuitively, if reading σ leads the current state p to state q, then the automaton can return to state p by scanning some string x. Therefore *retrievability* can also be viewed as a sort of reversibilities.

Indeed, reversibility associated with quantum finite automata is closely related to DFAs [13,2]. Brodsky and Pippenger [13] defined that a *group automaton* is a DFA $M = (Q, \Sigma, \delta)$ that for each given state $q \in Q$ and input symbol $\sigma \in \Sigma$, there exists exactly one state $q' \in Q$ such that $\delta(q', \sigma) = q$. Ambainis and Freivalds [2] defined *reversible finite automata* in terms of quantum finite automata with the restriction that the transition amplitudes are either 0 or 1; namely, according to [2], a finite automaton $M = (Q, \Sigma, \delta)$ is reversible if and only if M is a DFA and satisfies that, for any $q \in Q$ and every input symbol $\sigma \in \Sigma$, there exists at most one state $q' \in Q$ such that $\delta(q', \sigma) = q$. In this paper, we call the reversibility in the sense of [2] as *AF-reversibility*.

Since a reversible finite automaton and a group automaton, defined respectively by Brodsky and Pippenger [13], and Ambainis and Freivalds [2], which are concerned with quantum finite automata, are required to be DFAs, it is worth exactly clarifying their relationships. As indicated in [32], a language recognized by a reversible finite automaton may be accepted by a nonreversible one, for example, the corresponding minimal automaton is likely nonreversible. But here, as in [11,5], we focus on characterizing their algebraic properties. In the light of the reversibility proposed by Bavel and Muller [11], we call a finite automaton $M = (Q, \Sigma, \delta)$ to be *BM-reversible* if M is a DFA and satisfies that for any $\sigma \in \Sigma$ there exists string $x \in \Sigma^*$ such that for any $q \in Q$, $\delta(q, \sigma x) = q$. We now state our result as follows:

Proposition 3.1. *Let $M = (Q, \Sigma, \delta)$ be a DFA. Then the following three statements are equivalent.*

(i) *M is a group automaton, i.e. for any $q \in Q$ and any $\sigma \in \Sigma$, there exists unique $p \in Q$ such that $\delta(p, \sigma) = q$.*

(ii) *M is a BM-reversible automaton, i.e. for any $\sigma \in \Sigma$, there exists string $x \in \Sigma^*$ such that for any $q \in Q$, $\delta(q, \sigma x) = q$.*

(iii) *M is an AF-reversible automaton, i.e. for any $q \in Q$ and any $\sigma \in \Sigma$, there is at most one $p \in Q$ such that $\delta(p, \sigma) = q$.*

Proof. (i) ⇒ (ii): Assume that $Q = \{p_1, p_2, \ldots, p_m\}$. Since M is a group automaton, for any $\sigma \in \Sigma$, $\delta(\cdot, \sigma)$ is a bijection from Q to Q, i.e. a permutation on Q. Because the number of different arrangements for $p_1, p_2, \ldots, p_m$ is $m!$, there does exist k with $1 \leq k \leq m!$ such that $\delta(p_i, \sigma^k) = p_i$ for any $p_i \in Q$. Therefore, taking $x = \sigma^{k-1}$ results in (ii), as desired.

(ii) ⇒ (i): Since M is a DFA, it suffices to show that for any $\sigma \in \Sigma$ and any $p_1, p_2 \in Q$, $\delta(p_1, \sigma) = \delta(p_2, \sigma)$ implies $p_1 = p_2$. Indeed, by (ii) there exists $x \in \Sigma^*$ such that $\delta(q, \sigma x) = q$ for any $q \in Q$. Therefore, by combining $\delta(p_1, \sigma) = \delta(p_2, \sigma)$, we have that $p_1 = \delta(p_1, \sigma x) = \delta(p_2, \sigma x) = p_2$, as desired.

(i) ⇒ (iii): It is obvious.

(iii) ⇒ (i): By (iii) it suffices to show that for any $\sigma \in \Sigma$ and any $q \in Q$ there exists $p \in Q$ such that $\delta(p, \sigma) = q$. Assume that for any $p \in Q$, $\delta(p, \sigma) \neq q$. Then $\{\delta(p, \sigma) : p \in Q\} \subseteq Q \backslash \{q\}$. From M being a DFA it follows that there exist different states $p_1, p_2 \in Q$ such that $\delta(p_1, \sigma) = \delta(p_2, \sigma)$, which contradicts the condition (iii). Therefore (i) holds, and the proof is completed. □

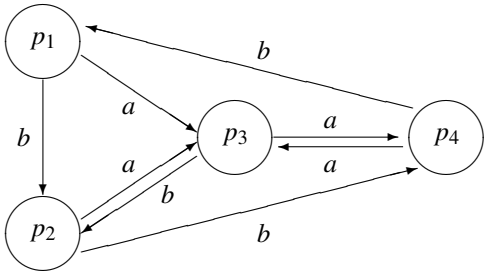

Fig. 1. A state transition diagram of finite automaton M_2 in Example 3.1.

Regarding the connection between AF-reversibility and retrievability, we have the following results.

Proposition 3.2. *If $M = (Q, \Sigma, \delta)$ is BM-reversible, then M is retrievable, that is, for any $p, q \in Q$ and any $\sigma \in \Sigma$, if $\delta(p, \sigma) = q$, then there exists $x \in \Sigma^*$ such that $\delta(q, x) = p$.*

Proof. Suppose that $\delta(p, \sigma) = q$. Since M is BM-reversible, there exists string $x \in \Sigma^*$ such that $\delta(p, \sigma x) = p$. If $x = \epsilon$, then $\delta(p, \sigma) = p$, and thus $p = q$ due to M being a DFA. In this case, we verify the desired result. Otherwise, we have $\delta(q, x) = p$, and the proof is completed. □

Remark 3.1. Nevertheless, a retrievable DFA is not necessarily BM-reversible. For example, consider finite automaton $M_1 = (Q, \Sigma, \delta)$, where $Q = \{p, q\}$, $\Sigma = \{\sigma_1, \sigma_2\}$, and δ is defined as: $\delta(p, \sigma_1) = q$; $\delta(q, \sigma_1) = q$; $\delta(q, \sigma_2) = p$; $\delta(p, \sigma_2) = q$. Then the automaton M_1 is clearly retrievable, and it is also a DFA but not AF-reversible (one can readily describe the transition diagram of this finite automaton). Another finite automaton M_2, described in the following Example 3.1, also verifies this view.

Example 3.1. Let $M_2 = (Q, \Sigma, \delta)$ be a finite automaton, where $Q = \{p_1, p_2, p_3, p_4\}$, $\Sigma = \{a, b\}$, and its transition diagram is depicted by Fig. 1. Then M_2 is a retrievable DFA, but clearly it is not AF-reversible since $\delta(p_1, a) = \delta(p_2, a) = p_3$.

3.2. Reversibilities of l-valued finite automata

By Remark 3.1 and Example 3.1 we know that a retrievable DFA is not necessarily a group automaton, so, in the aforementioned various reversibilities, retrievability is the weakest one. However, it is worth stressing that a retrievable finite automaton, even if it is not a DFA, still has the ability to retrieve a state after leaving it, which is an important property of finite automata [11,5]. Therefore, in this subsection, we deal with retrievability of l-VFAs. First we introduce a related concept—*separability*. A subautomaton of a finite automaton is said to be separated, if the complement of its set of states (equipped with the restriction of the original transition relation) is also a subautomaton of the automaton. This concept may be clearly generalized into the setting of l-VFAs.

Definition 3.1. Let $\mathcal{M} = (Q, \Sigma, \delta)$ be an l-VFA, and let $A \in L^Q$. Then A is said to be separated if $\overset{l}{\models} S(A^\perp) \subseteq A^\perp$.

The separability can also be characterized in terms of the source operator R.

Proposition 3.3. *The following two statements are equivalent:*
(i) *L is a Boolean algebra.*
(ii) *For any l-VFA $\mathcal{M} = (Q, \Sigma, \delta)$, and for any $A \in L^Q$, then A is separated iff $\overset{l}{\models} R(A) \subseteq A$.*

Proof. Immediate from Proposition 2.1. □

Two sufficient conditions for the separability are given in the following proposition under the assumption of the distributivity.

Proposition 3.4. *Let $\mathcal{M} = (Q, \Sigma, \delta)$ be an l-VFA. Suppose that L is a Boolean algebra. Then*

(i) *if $\overset{l}{\models} S(R(A)) \equiv R(A)$, then $R(A)$ is a separated l-valued subautomaton of $\mathcal{M}$;*

(ii) *if $\overset{l}{\models} R(S(A)) \equiv S(A)$, then $S(A)$ is a separated l-valued subautomaton of $\mathcal{M}$.*

Proof. (i) By Proposition 2.1, we see that $R(A)$ is an l-valued subautomaton of $\mathcal{M}$. Moreover, it follows from Proposition 2.2(iii) that $\overset{l}{\models} R(R(A)) \equiv R(A)$. With Proposition 2.1 again we obtain that $\overset{l}{\models} S(R(A)^{\perp}) = R(A)^{\perp}$, and hence $R(A)$ is a separated l-valued subautomaton of $\mathcal{M}$.

(ii). Similar to (i). □

Now we introduce a counterpart of retrievability in the framework of l-VFAs.

Definition 3.2. Let $\mathcal{M} = (Q, \Sigma, \delta)$ be an l-VFA. Then $\mathcal{M}$ is said to be *retrievable* if for any $p, q \in Q$ and any $x \in \Sigma^*$,

$$\overset{l}{\models} (p, x, q) \in \delta^* \longrightarrow (\exists y \in \Sigma^*)((q, y, p) \in \delta^*),$$

that is, for any $p, q \in Q$ and any $x \in \Sigma^*$,

$$\delta^*(p, x, q) \leq \bigvee_{y \in \Sigma^*} \delta^*(q, y, p). \tag{1}$$

The notion of retrievability may be further generalized to l-valued subautomata: an l-valued subautomaton A of $\mathcal{M}$ is retrievable if for any $p, q \in Q$ and for any $x \in \Sigma^*$,

$$\overset{l}{\models} p \in A \longrightarrow ((p, x, q) \in \delta^* \to (\exists y \in \Sigma^*)((q, y, p) \in \delta^* \wedge q \in A)),$$

that is,

$$A(p) \& \delta^*(p, x, q) \leq \left(\bigvee_{y \in \Sigma^*} \delta^*(q, y, p) \right) \wedge A(q).$$

For any l-VFA $\mathcal{M} = (Q, \Sigma, \delta)$, let us consider the following seven statements:

(I) $\mathcal{M}$ is retrievable.

(II) For any $p, q \in Q$ and for any $\sigma \in \Sigma$,

$$\overset{l}{\models} (p, \sigma, q) \in \delta \longrightarrow (\exists y \in \Sigma^*)((q, y, p) \in \delta^*).$$

(III) For any l-valued subautomaton A of $\mathcal{M}$, $\overset{l}{\models} R(A) \equiv A$.

(IV) For any $p, q \in Q$, $\overset{l}{\models} q \in S(\{p\}) \to p \in S(\{q\})$.

(V) For any $p, q \in Q$, $\overset{l}{\models} q \in R(\{p\}) \to p \in R(\{q\})$.

(VI) For any l-subautomaton A of $\mathcal{M}$, A is retrievable.

(VII) For any l-subautomaton A of $\mathcal{M}$, A is separated.

Remark 3.2. In the sense of classical finite automata [11,5], the above seven statements are mutually equivalent. Nevertheless, in l-VFAs, their relationships are more complicated, and the following series of claims clarifies these relationships, which also reveal some essential differences between classical finite automata and l-VFAs.

Let us list these claims and verify them.

Claim 1. **(I)** ⇒ **(II)**.

Proof. Obvious. □

Claim 2. *If L satisfies the infinite distributive law of $\wedge$ over $\vee$, i.e., $a \wedge (\vee_i b_i) = \vee_i (a \wedge b_i)$ for any $a, b_i \in L$, then* **(II) ⇒ (I)***; conversely, if for any l-VFA $\mathcal{M} = (Q, \Sigma, \delta)$,* **(II) ⇒ (I)***, then L satisfies the distributive law of $\wedge$ over $\vee$, i.e. $a \wedge (b \vee c) = (a \wedge b) \vee (a \wedge c)$ for any $a, b, c \in L$.*

Proof. First suppose that L satisfies the infinite distributivity. Our purpose is to show that Ineq. (1) holds for any $p, q \in Q$ and any $x \in \Sigma^*$. We proceed by induction on the length of x. It is clear for the cases of $|x| = 1$ and $|x| = 0$. (In this paper, $|x|$ denotes the length of string x.) Assume that the conclusion holds for $|x| \leq k - 1$, and $x = \sigma_1\sigma_2 \cdots \sigma_k \in \Sigma^*$, $\sigma_i \in \Sigma$. By **(II)** and the induction hypothesis we have

$$\begin{aligned}
\delta^*(p, \sigma_1\sigma_2 \cdots \sigma_k, q) &= \bigvee_{r \in Q} (\delta^*(p, \sigma_1\sigma_2 \cdots \sigma_{k-1}, r) \wedge \delta(r, \sigma_k, q)) \\
&\leq \bigvee_{r \in Q} \left(\left(\bigvee_{y_1 \in \Sigma^*} \delta^*(r, y_1, p) \right) \wedge \left(\bigvee_{y_2 \in \Sigma^*} \delta^*(q, y_2, r) \right) \right) \\
&= \bigvee_{r \in Q} \bigvee_{y_1 \in \Sigma^*} \bigvee_{y_2 \in \Sigma^*} (\delta^*(q, y_2, r) \wedge \delta^*(r, y_1, p)) \\
&= \bigvee_{y_1 \in \Sigma^*} \bigvee_{y_2 \in \Sigma^*} \delta^*(q, y_2 y_1, p) \\
&= \bigvee_{y \in \Sigma^*} \delta^*(q, y, p).
\end{aligned}$$

So inequality (1) holds, and it yields that **(II) ⇒ (I)**.

Conversely, for any $a, b, c \in L$, let us set $\mathcal{M} = (Q, \Sigma, \delta)$, where $Q = \{p, q, r_1, r_2, r_3\}$, $\Sigma = \{\sigma_1, \sigma_2, \sigma_3\}$, $\delta(p, \sigma_1, r_2) = \delta(r_2, \sigma_2, r_1) = b$, $\delta(p, \sigma_1, r_3) = \delta(r_3, \sigma_2, r_1) = c$, $\delta(r_1, \sigma_3, q) = \delta(q, \sigma_1, r_1) = a$, $\delta(r_1, \sigma_2, r_2) = \delta(r_2, \sigma_3, p) = b$, $\delta(r_1, \sigma_2, r_3) = \delta(r_3, \sigma_3, p) = c$. It is easy to check that for any $r', r'' \in Q$ and for any $\sigma \in \Sigma$,

$$\delta(r', \sigma, r'') \leq \bigvee_{x \in \Sigma^*} \delta(r'', x, r'). \tag{2}$$

With a simple calculation it follows that

$$\begin{aligned}
\delta^*(p, \sigma_1\sigma_2\sigma_3, q) &= \delta^*(p, \sigma_1\sigma_2, r_1) \wedge \delta^*(r_1, \sigma_3, q) \\
&= ((\delta(p, \sigma_1, r_2) \wedge \delta(r_2, \sigma_2, r_1)) \vee (\delta(p, \sigma_1, r_3) \wedge \delta(r_3, \sigma_2, r_1))) \wedge \delta(r_1, \sigma_3, q) \\
&= (b \vee c) \wedge a,
\end{aligned}$$

and

$$\begin{aligned}
\bigvee_{y \in \Sigma^*} \delta^*(q, y, p) &= \delta^*(q, \sigma_1\sigma_2\sigma_3, p) \\
&= (\delta^*(q, \sigma_1\sigma_2, r_2) \wedge \delta(r_2, \sigma_3, p)) \vee (\delta^*(q, \sigma_1\sigma_2, r_3) \wedge \delta(r_3, \sigma_3, p)) \\
&= (\delta(q, \sigma_1, r_1) \wedge \delta(r_1, \sigma_2, r_2) \wedge \delta(r_2, \sigma_3, p)) \vee (\delta(q, \sigma_1, r_1) \wedge \delta(r_1, \sigma_2, r_3) \wedge \delta(r_3, \sigma_3, p)) \\
&= (a \wedge b \wedge b) \vee (a \wedge c \wedge c) \\
&= (a \wedge b) \vee (a \wedge c).
\end{aligned}$$

So with Ineq. (2) we obtain that $a \wedge (b \vee c) \leq (a \wedge b) \vee (a \wedge c)$, and further $a \wedge (b \vee c) = (a \wedge b) \vee (a \wedge c)$ because $a \wedge (b \vee c) \geq (a \wedge b) \vee (a \wedge c)$ is always valid. □

Claim 3. (I) ⇒ (IV).

Proof. It is enough to verify that for any $p, q \in Q$,

$$S(\{p\})(q) \leq S(\{q\})(p). \tag{3}$$

Notice that

$$S(\{p\})(q) = \bigvee_{x \in \Sigma^*} \delta^*(p, x, q) \tag{4}$$

and

$$S(\{q\})(p) = \bigvee_{y\in\Sigma^*} \delta^*(q, y, p). \tag{5}$$

By combining Ineq. (1) with Eqs. (4) and (5) it is easy to see that Ineq. (3) holds. □

Claim 4. (IV) ⇒ (I).

Proof. Straightforward from Ineq. (3), Eqs. (4) and (5). □

Claim 5. (I) ⇔ (V).

Proof. Similar to the proofs of **(I) ⇒ (IV)** and **(IV) ⇒ (I)**. □

Claim 6. (I) ⇒ (VI).

Proof. It suffices to show that for any $p, q \in Q$ and any $x \in \Sigma^*$,

$$A(p) \le \delta^*(p, x, q) \longrightarrow \left(\bigvee_{y\in\Sigma^*} \delta^*(q, y, p)\right) \wedge A(q); \tag{6}$$

with **(4.1)** in Section 2 it is equivalent to

$$A(p)\&\delta^*(p, x, q) \le \left(\bigvee_{y\in\Sigma^*} \delta^*(q, y, p)\right) \wedge A(q). \tag{7}$$

Since A is an l-valued subautomaton of $\mathcal{M}$, we have

$$A(p) \le \delta^*(p, x, q) \to A(p),$$

equivalently,

$$A(p)\&\delta^*(p, x, q) \le A(q). \tag{8}$$

On the other hand, by **(I)** one has

$$\delta^*(p, x, q) \le \bigvee_{y\in\Sigma^*} \delta^*(q, y, p),$$

and with **(4.6)** in Section 2

$$A(p)\&\delta^*(p, x, q) \le \delta^*(p, x, q) \le \bigvee_{y\in\Sigma^*} \delta^*(q, y, p). \tag{9}$$

By combining Ineqs. (8) and (9) one may assert Ineq. (7) which yields Ineq. (6). □

Claim 7. (VI) ⇒ (I).

Proof. It suffices to note that $\mathcal{M}$ itself is an l-valued subautomaton of $\mathcal{M}$. □

Claim 8. *If L satisfies the infinite distributivity of $\wedge$ over $\vee$, then* **(I) ⇒ (VII).**

Proof. For any $p \in Q$, by **(I)** one has

$$\begin{aligned} S(A^\perp)(p) &= \bigvee_{q\in Q} \bigvee_{x\in\Sigma^*} (\delta^*(q, x, p) \wedge A^\perp(q)) \\ &\le \bigvee_{q\in Q} \bigvee_{x\in\Sigma^*} \left(\bigvee_{y\in\Sigma^*} \delta^*(p, y, q) \wedge A^\perp(q)\right). \end{aligned}$$

On the other hand, since A is an l-valued subautomaton of $\mathcal{M}$, we know that

$$A(p)\&\delta^*(p, y, q) \le A(q).$$

By *De Morgan* law we have

$$\begin{aligned} A(q)^\perp &\le (A(p)\&\delta^*(p, y, q))^\perp \\ &= \delta^*(p, y, q) \to A(p)^\perp. \end{aligned}$$

Thus, by **(4.7)**, **(4.8)** and **(4.4)** in Section 2, we obtain

$$\begin{aligned} \delta^*(p, y, q) \wedge A(q)^\perp &\le \delta^*(p, y, q)\&A(q)^\perp \\ &\le \delta^*(p, y, q)\&(\delta^*(p, y, q) \to A(p)^\perp) \\ &\le A(p)^\perp. \end{aligned}$$

Using the infinite distributivity, we obtain that for any $q \in Q$ and $x \in \Sigma^*$,

$$\left(\bigvee_{y\in\Sigma^*} \delta^*(p, y, q)\right) \wedge A(q)^\perp = \bigvee_{y\in\Sigma^*} (\delta^*(p, y, q) \wedge A(q)^\perp) \le A(p)^\perp,$$

and $S(A^\perp)(p) \le \bigvee_{q\in Q} \bigvee_{x\in\Sigma^*} A(p)^\perp = A(p)^\perp$. Thus it holds that $\overset{l}{\models} S(A^\perp) \equiv A^\perp$. □

Claim 9. *If L satisfies the infinite distributive law, then* **(I)⇔(III)** *and* **(VII)⇒(I)**.

Proof. **(I)⇒(III)**: It suffices to show that for any $p \in Q$, $R(A)(p) \le A(p)$. By **(I)** we have

$$\begin{aligned} R(A)(p) &= \bigvee_{x\in\Sigma^*} \bigvee_{q\in Q} (\delta^*(p, x, q) \wedge A(q)) \\ &\le \bigvee_{x\in\Sigma^*} \bigvee_{q\in Q} \left(\left(\bigvee_{y\in\Sigma^*} \delta^*(q, y, p)\right) \wedge A(q)\right) \\ &= \bigvee_{q\in Q} \bigvee_{y\in\Sigma^*} (\delta^*(q, y, p) \wedge A(q)) \\ &= S(A)(p). \end{aligned}$$

Since A is an l-valued subautomaton of $\mathcal{M}$, $S(A)(p) \le A(p)$. This yields that $R(A)(p) \le A(p)$.

(III)⇒(I): Our aim is to show that Ineq. (1) holds for any $p, q \in Q$ and any $x \in \Sigma^*$. By Proposition 2.1 we know that $S(\{q\})$ is an l-valued subautomaton of $\mathcal{M}$. From **(III)** it follows that

$$\overset{l}{\models} R(S(\{q\})) \equiv S(\{q\}).$$

Consequently,

$$R(S(\{q\}))(p) = S(\{q\})(p). \tag{10}$$

By the definitions of S and R, one has

$$S(\{q\})(p) = \bigvee_{y\in\Sigma^*} \delta^*(q, y, p)$$

and

$$\begin{aligned} R(S(\{q\}))(p) &= \bigvee_{r\in Q} \bigvee_{z\in\Sigma^*} \left(\delta^*(p, z, r) \wedge \left(\bigvee_{w\in\Sigma^*} \delta^*(q, w, r)\right)\right) \\ &\ge \delta^*(p, x, q). \end{aligned}$$

Therefore, by Eq. (10) it follows that

$$\delta^*(p, x, q) \leq \bigvee_{y \in \Sigma^*} \delta^*(q, y, p).$$

(VII)⇒(I): By Proposition 2.1 we know that $\overset{l}{\models} S(A^\perp) \equiv A^\perp$ iff $\overset{l}{\models} R(A) \equiv A$ since L is assumed to satisfy the distributive law. This tells us that **(VII)⇒(III)**. By noting **(III)⇒(I)** we complete the proof. □

By summarizing the above nine claims, we have the following two theorems.

Theorem 3.5. *Let $\mathcal{M} = (Q, \Sigma, \delta)$ be an l-VFA. Then* **(I)**, **(IV)**, **(V)**, *and* **(VI)** *are mutually equivalent.*

Proof. Immediate from Claims 3–7. □

Theorem 3.6. *If L satisfies the infinite distributive law of $\wedge$ over $\vee$, then* **(I)**–**(VII)** *are mutually equivalent.*

Proof. Immediate from Theorem 3.5 and Claims 1, 2, 8 and 9. □

4. l-valued pushdown automata

Before defining l-VPDAs, we briefly recall classical pushdown automata (abbr. PDAs), and the details are referred to [25]. A PDA M is a system $(Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$, where: (1) Q is a finite set of states; (2) Σ is a finite input alphabet; (3) Γ is a finite stack alphabet; (4) $q_0 \in Q$ is the initial state; (5) $Z_0 \in \Gamma$ is the start stack symbol; (6) $F \subseteq Q$ is the set of final states; (7) δ is a mapping from $Q \times (\Sigma \cup \{\epsilon\}) \times \Gamma$ to $\mathcal{P}(Q \times \Gamma^*)$. (In this paper, $\mathcal{P}(X)$ denotes the power set of X.)

For any $(q, a, Z) \in Q \times \Sigma \times \Gamma$ and $(p, \gamma) \in Q \times \Gamma^*$, $(p, \gamma) \in \delta(q, a, Z)$ represents that the PDA, in state q with input symbol a and the top stack symbol Z, can enter state p and replace symbol Z by string γ as well as advance the input head one symbol, while the interpretation of $(p, \gamma) \in \delta(q, \epsilon, Z)$ is that the PDA, in state q and the top stack symbol Z, can enter state p and substitute γ for Z, without scanning any input symbol, and the head is not advanced.

As it was known, there are two fashions for defining the languages accepted by PDAs: one is by final state and the other by empty stack. More specifically, an input symbol string $w \in \Sigma^*$ is accepted by PDA M above by final state, if and only if the machine M, in initial state q_0, scans w in sequence and then enters a final state; and $w \in \Sigma^*$ is accepted by the M by empty stack if and only if the machine M, in initial state q_0, scans w in sequence and finally its stack is emptied.

Now we give the definition of l-VPDAs.

Definition 4.1. An l-VPDA is defined as a system $M = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$, where Q, Σ, Γ, δ, q_0, Z_0, and F are the same as in classical pushdown automata defined above, and δ is defined as a mapping from $Q \times (\Sigma \cup \{\epsilon\}) \times \Gamma \times Q \times \Gamma^*$ to L.

Intuitively, for any $p, q \in Q$, $\sigma \in \Sigma \cup \{\epsilon\}$, $Z \in \Gamma$, and $\alpha \in \Gamma^*$, $\delta(p, \sigma, Z, q, \alpha)$ represents the degree to which the machine, in the current state p and the top stack symbol Z, enters state q and the top stack symbol Z is replaced by string α, after scanning σ (if $\sigma = \epsilon$, then no input symbol is scanned).

A *configuration* (or called *instantaneous description*) of l-VPDA M is defined to be a triple (q, w, γ), where $q \in Q$, $w \in \Sigma^*$, and $\gamma \in \Gamma^*$. We denote by $\mathcal{C}(M)$ the set of all configurations of M, and by $\overset{M}{\vdash}$ we define a map from $\mathcal{C}(M) \times \mathcal{C}(M)$ to L: for any configurations (q_1, w_1, γ_1) and (q_2, w_2, γ_2), where $w_1 = w_1^{(1)} w_1^{(2)} \cdots w_1^{(|w_1|)} \in \Sigma^*$, $\gamma_1 = \gamma_1^{(1)} \gamma_1^{(2)} \cdots \gamma_1^{(|\gamma_1|)} \in \Gamma^*$, $w_1^{(i)} \in \Sigma$, $\gamma_1^{(j)} \in \Gamma$,

$$\overset{M}{\vdash} [(q_1, w_1, \gamma_1), (q_2, w_2, \gamma_2)] = \begin{cases} \delta(q_1, w_1^{(1)}, \gamma_1^{(1)}, q_2, \gamma), & \text{if } w_2 = w_1^{(2)} w_1^{(3)} \cdots w_1^{(|w_1|)} \\ & \text{and } \gamma_2 = \gamma \gamma_1^{(2)} \gamma_1^{(3)} \cdots \gamma_1^{(|\gamma_1|)}, \\ \delta(q_1, \epsilon, \gamma_1^{(1)}, q_2, \gamma), & \text{if } w_1 = w_2 \\ & \text{and } \gamma_2 = \gamma \gamma_1^{(2)} \gamma_1^{(3)} \cdots \gamma_1^{(|\gamma_1|)}, \\ 0, & \text{otherwise.} \end{cases} \tag{11}$$

Furthermore, we extend $\overset{M}{\vdash}$ to $\overset{M*}{\vdash}$, by defining that for any $C, C' \in \mathcal{C}(M)$,

$$\overset{M*}{\vdash} [C, C'] = \bigvee \left\{ \overset{M}{\vdash} [C, C_1] \wedge \bigwedge_{i=1}^{m} \overset{M}{\vdash} [C_i, C_{i+1}] : m \in N, C_i \in \mathcal{C}(M), C_{m+1} = C', i = 1, 2, \ldots, m \right\}, \quad (12)$$

where N denotes the set of all natural numbers. For simplicity, the superscript M is sometimes dropped from $\overset{M}{\vdash}$ and $\overset{M*}{\vdash}$ if no confusion results.

As in classical case, we define the languages accepted by l-VPDAs via two fashions.

Definition 4.2. Let $M = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ be an l-VPDA. Then the languages g_M^T and g_M^S accepted by M via final state and via empty stack, respectively, are defined as two mappings from Σ^* to L: for any $w = w_1 w_2 \cdots w_n \in \Sigma^*$, $w_i \in \Sigma$,

$$g_M^T(w) = \bigvee \left\{ \overset{*}{\vdash} [(q_0, w, Z_0), (q, \epsilon, \gamma)] : q \in F, \gamma \in \Gamma^* \right\}; \quad (13)$$

$$g_M^S(w) = \bigvee \left\{ \overset{*}{\vdash} [(q_0, w, Z_0), (q, \epsilon, \epsilon)] : q \in Q \right\}. \quad (14)$$

In classical PDAs, the class of the languages accepted by PDAs by final state accords with that accepted by PDAs by empty stack. We show that this result is reserved in the setting of l-VPDAs.

Theorem 4.1. *Let $M = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ be an l-VPDA. Then there exists l-VPDA M' such that for any $w \in \Sigma^*$,*

$$g_M^T(w) = g_{M'}^S(w). \quad (15)$$

Proof. The basic idea of the proof is similar to classical case (please refer to Theorem 5.3 in [25]). First we construct $M' = (Q', \Sigma, \Gamma', \delta', q_0', Z_0', \emptyset)$, where $Q' = Q \cup \{q_0', q_t\}$, $\Gamma' = \Gamma \cup \{Z_0'\}$, and $q_0', q_t \notin Q$, $Z_0' \notin \Gamma$; δ' is defined as follows:

$$\delta'(q_0', \epsilon, Z_0', q_0, Z_0 Z_0') = 1; \quad (16)$$

for any $p, q \in Q$, $\sigma \in \Sigma \cup \{\epsilon\}$, any $Z \in \Gamma$, and any $\alpha \in \Gamma^*$,

$$\delta'(q, \sigma, Z, p, \alpha) = \delta(q, \sigma, Z, p, \alpha); \quad (17)$$

for any $q \in F$, $Z \in \Gamma \cup \{Z_0'\}$,

$$\delta'(q, \epsilon, Z, q_t, \epsilon) = \delta'(q_t, \epsilon, Z, q_t, \epsilon) = 1. \quad (18)$$

To show that Eq. (15), we first show that

$$g_M^T(w) \le g_{M'}^S(w). \quad (19)$$

For any $q \in F$, $\gamma \in \Gamma^*$, $q_i \in Q$, and $x_i \in \Sigma^*$, $\gamma_i \in \Gamma^*$, $i = 1, 2, \ldots, k$, where $x_k = \epsilon$ and $q_k = q$, by the definitions of $g_{M'}^S(w)$ and δ', we have

$$\begin{aligned} g_{M'}^S(w) \ge\ & \overset{M'}{\vdash} [(q_0', w, Z_0'), (q_0, w, Z_0 Z_0')] \wedge \overset{M'}{\vdash} [(q_0, w, Z_0 Z_0'), (q_1, x_1, \gamma_1 Z_0')] \\ & \wedge \bigwedge_{i=1}^{k-1} \overset{M'}{\vdash} [(q_i, x_i, \gamma_i Z_0'), (q_{i+1}, x_{i+1}, \gamma_{i+1} Z_0')] \wedge \overset{M'*}{\vdash} [(q, \epsilon, \gamma_k Z_0'), (q_t, \epsilon, Z_0')] \\ & \wedge \overset{M'}{\vdash} [(q_t, \epsilon, Z_0'), (q_t, \epsilon, \epsilon)]. \end{aligned} \quad (20)$$

Note that with Eq. (16)

$$\overset{M'}{\vdash} [(q_0', w, Z_0'), (q_0, w, Z_0 Z_0')] = \delta'(q_0', \epsilon, Z_0', q_0, Z_0 Z_0') = 1; \quad (21)$$

and with Eqs. (18) and (17)

$$\overset{M'*}{\vdash}[(q,\epsilon,\gamma_k Z_0'),(q_t,\epsilon,Z_0')]\wedge\overset{M'}{\vdash}[(q_t,\epsilon,Z_0'),(q_t,\epsilon,\epsilon)]=1; \tag{22}$$

$$\overset{M'}{\vdash}[(q_0,w,Z_0Z_0'),(q_1,x_1,\gamma_1Z_0')]\wedge\bigwedge_{i=1}^{k-1}\overset{M'}{\vdash}[(q_i,x_i,\gamma_iZ_0'),(q_{i+1},x_{i+1},\gamma_{i+1}Z_0')]$$

$$=\overset{M}{\vdash}[(q_0,w,Z_0),(q_1,x_1,\gamma_1)]\wedge\bigwedge_{i=1}^{k-1}\overset{M}{\vdash}[(q_i,x_i,\gamma_i),(q_{i+1},x_{i+1},\gamma_{i+1})]. \tag{23}$$

By combining Ineq. (20) with Eqs. (21), (22) and (23), we obtain that

$$g^S_{M'}(w)\ \geq\ \overset{M'}{\vdash}[(q_0,w,Z_0Z_0'),(q_1,x_1,\gamma_1Z_0')]\wedge\bigwedge_{i=1}^{k-1}\overset{M'}{\vdash}[(q_i,x_i,\gamma_iZ_0'),(q_{i+1},x_{i+1},\gamma_{i+1}Z_0')]$$

$$=\overset{M}{\vdash}[(q_0,w,Z_0),(q_1,x_1,\gamma_1)]\wedge\bigwedge_{i=1}^{k-1}\overset{M}{\vdash}[(q_i,x_i,\gamma_i),(q_{i+1},x_{i+1},\gamma_{i+1})]. \tag{24}$$

Therefore we have shown that Ineq. (19) holds, i.e., $g^T_M(w)\leq g^S_{M'}(w)$. In the remainder we prove the converse inequality. By means of the definition of δ' we exactly have

$$g^S_{M'}(w)\ =\bigvee\left\{\overset{M'}{\vdash}[(q_0',w,Z_0'),(q_0,w,Z_0Z_0')]\wedge\overset{M'*}{\vdash}[(q_0,w,Z_0Z_0'),(q,\epsilon,\gamma Z_0')]\right.$$

$$\left.\wedge\overset{M'*}{\vdash}[(q_0,\epsilon,\gamma Z_0'),(q_t,\epsilon,\epsilon)]:q\in F,\gamma\in\Gamma^*\right\}. \tag{25}$$

With the definition of $g^T_M(w)$ it is seen that

$$\overset{M'*}{\vdash}[(q_0,w,Z_0Z_0'),(q,\epsilon,\gamma Z_0')]\leq g^T_M(w). \tag{26}$$

By Eq. (25) and Ineq. (26) we thus obtain that $g^T_M(w)\geq g^S_{M'}(w)$. Therefore, we have demonstrated $g^T_M(w)=g^S_{M'}(w)$. □

Theorem 4.2. *Let $M=(Q,\Sigma,\Gamma,\delta,q_0,Z_0,\emptyset)$ be an l-VPDA. Then there exists l-VPDA M_s such that for any $w\in\Sigma^*$,*

$$g^T_{M_s}(w)=g^S_M(w). \tag{27}$$

Proof. Also, the idea is similar to classical situation (Theorem 5.2 in [25]). Let us first construct the l-VPDA $M_s=(Q_s,\Sigma,\Gamma_s,\delta_s,q_0',Z_0',\{q_0\})$, where $Q_s=Q\cup\{q_0',q_s\}$, and $q_0',q_s\notin Q$; $\gamma_s=\gamma\cup\{Z_0'\}$; $Z_0'\notin\Gamma$; δ_s is defined as follows:

$\delta_s(q_0',\epsilon,Z_0',q_0,Z_0Z_0')=1$;

and for any $p,q\in Q,\sigma\in\Sigma\cup\{\epsilon\},Z\in\Gamma,\alpha\in\Gamma^*$,

$$\delta_s(q,\sigma,Z,p,\alpha)=\delta(q,\sigma,Z,p,\alpha); \tag{28}$$

as well as for any $q\in Q$, $\delta_s(q,\epsilon,Z_0',q_s,\epsilon)=1$. The rest of the proof is exactly similar to Theorem 4.1, and we leave out the details here. □

5. Some concrete examples related to l-VFAs

In the previous sections, we gave the equivalence relations between some propositions in l-VFAs, and we found that some equivalence relations are independent of the distributivity of the truth-value lattice of the underlying logic, but for the others, they rely on the distributivity of logic. However, we note that those equivalence relations stress the

connections between the *distributivity* and some properties "*for any*" l-VFAs. Then we may raise whether there are *some* l-VFAs such that those properties still hold for (*nondistributive*) l-valued logics. In this section, we present some examples to verify this viewpoint.

As it is well known, classical finite automata [25] are simple but important models of computation and have been applied to many fields such as programming languages, compiler constructions, lexical analysis, description of natural languages and other practical issues (for example, discrete event systems). Regarding recent applications and development of finite automata, we refer to [26] and the references therein. From the logical point of view, computation processes in classical computers obey classical logic, i.e. Boolean logic. However, in a quantum computer, fundamental computation processes comply with the rules of quantum mechanics, and therefore, we may be led to considering computation theory based on quantum logic that was regarded as describing quantum physical systems in the early of 1930s by von Neumann and Birkhoff [12].

Quantum logic, as indicated above, was proposed for investigating formally the differences between quantum mechanics and classical mechanics. The starting point of this study is based on the use of closed subspaces of a Hilbert space to represent those propositions about the system under consideration. Built on the operations defined on closed subspaces, including orthocomplement and intersection, that are respectively interpreted as negation and conjunction on propositions, the orthomodular lattice-valued logic is usually thought of as standard quantum logic [33]. In recent years, with the intensive interest in quantum computing in the academic community, quantum logic has been developed and given preliminary applications to some areas in quantum computing, such as quantum information processing [6], quantum logical gates [15], quantum programs [10], and quantum computational logics [18]. The relation of quantum logic with Bell inequalities was described in [42].

In the interest of readability, we recall the syntax and semantics of l-valued logic again, that were indeed described in Section 2.1. Let $l = \langle L, \leq, \wedge, \vee, \bot, 0, 1\rangle$ be a complete lattice and $\rightarrow$ the Sasaki arrow. Regarding the syntax, there are three primitive connectives $\neg$ (negation), $\wedge$ (conjunction), $\rightarrow$ (implication) and a primitive quantifier $\forall$ (universal quantifier). Then, the connectives $\vee$ (disjunction) and $\leftrightarrow$ (bi-implication) and the existential quantifier $\exists$ are defined in terms of $\neg$, $\wedge$, $\rightarrow$ and $\forall$ in the usual way. Furthermore, suppose that $\in$ (membership) is a binary (primitive) predicate symbol. Then $\subseteq$ (inclusion) and $\equiv$ (equality) can be defined with $\in$ as usual. The semantics of l-valued logic is given by interpreting the connectives $\neg$, $\wedge$ and $\rightarrow$ as the operations $\bot$, $\wedge$ and $\rightarrow$, on L, respectively, and interpreting the quantifier $\forall$ as the greatest lower bound in L. In addition, the truth value of $x \in A$ is $\lceil x \in A\rceil = A(x)$; 1 is the unique designated truth value. A formula φ is valid iff $\lceil\varphi\rceil = 1$.

Theoretically, quantum computing is closely related to quantum systems with finite dimensional Hilbert spaces, especially the n-qubit state space $\mathbf{C}^{2^n} = \otimes^n\mathbf{C}^2$ [30,24] ($\mathbf{C}$ denotes the set of complex numbers). Thus we consider the set L_n—all closed subspaces of Hilbert space $\otimes^n\mathbf{C}^2$. It can be checked that $\langle L_n, \leq, \wedge, \vee, 0, 1\rangle$ is a complete lattice with set inclusion as the partial order relation $\leq$, and for any $M \subseteq L$, the meet $\wedge_{A\in M}A$ defined as the set intersection, the join $\vee_{A\in M}A$ defined as the closure of the subspace spanned by $\cup_{A\in M}A$, and $\{\mathbf{0}\}$ and $\otimes^n\mathbf{C}^2$ designated as 0 and 1, respectively, where $\mathbf{0}$ denotes the zero-vector of $\otimes^n\mathbf{C}^2$. Furthermore, by defining $A^\perp$ to be the orthocomplement of A, it is also clear to verify that $l_n = \langle L_n, \leq, \wedge, \vee, \bot, 0, 1\rangle$ is a complete orthomodular lattice.

Now we deal with some properties in l_2-VFAs, in which the truth value $\lceil p\rceil$ of each proposition p is identified with some element in L_2, i.e. a closed subspace of $\otimes^2\mathbf{C}^2$. Implication $\rightarrow$ is Sasaki arrow, that is, for any propositions p and q,

$$\lceil p \rightarrow q\rceil = \lceil \neg p \vee (p \wedge q)\rceil = \lceil p\rceil^\perp \vee (\lceil p\rceil \wedge \lceil q\rceil).$$

As the standard notation in quantum computation [30,24], $|0\rangle|0\rangle$, $|0\rangle|1\rangle$, $|1\rangle|0\rangle$, and $|1\rangle|1\rangle$ are four *computational basis states* in the 2-qubit state space $\otimes^2\mathbf{C}^2$. For convenience, we denote by $v_{ij} = \text{span}\{|i\rangle|j\rangle\}$ the closed subspace spanned by $|i\rangle|j\rangle$, $i, j = 0, 1$.

We define an l_2-VFA $\mathcal{M} = (Q, \Sigma, \delta)$ as follows: $Q = \{p, q\}$, $\Sigma = \{\sigma\}$, and δ is defined as

$$\begin{aligned} &\delta(p, \sigma, q) = v_{00}, \quad \delta(q, \sigma, p) = v_{11},\\ &\delta(p, \sigma, p) = v_{01}, \quad \delta(q, \sigma, q) = v_{10}. \end{aligned}$$

Furthermore, we define $A \in L_2^Q$ as follows:

$$A(p) = v_{11}, \quad A(q) = v_{00}.$$

Then we can verify that, by Definition 2.3 in Section 2, A is an l-valued subautomaton of $\mathcal{M}$. Indeed, first we note that

$$\delta^*(p,\sigma\sigma,p)=\delta(p,\sigma,p),\quad \delta^*(p,\sigma\sigma,q)=0,$$
$$\delta^*(q,\sigma\sigma,q)=\delta(q,\sigma,q),\quad \delta^*(q,\sigma\sigma,p)=0.$$

Then, furthermore, it is easy to check that for any states $r_1, r_2 \in Q$, and any $x \in \Sigma^k$ with $k \geq 1$,

$$\delta^*(r_1,x,r_2)=\begin{cases}\delta(r,\sigma,r), & \text{if } r_1=r_2=r\in Q,\\ 0, & \text{otherwise.}\end{cases}$$

Therefore, we have

$$\begin{aligned}
&\bigwedge\{\lceil\delta^*(p,x,r)\to A(r)\rceil : x\in\Sigma^*, r\in Q\}\\
&=\bigwedge\{\lceil\delta^*(p,x,q)\to A(q)\rceil : x\in\Sigma^*\}\wedge\bigwedge\{\lceil\delta^*(p,x,p)\to A(p)\rceil : x\in\Sigma^*\}\\
&=\lceil 0\to A(q)\rceil\wedge\lceil\delta(p,\sigma,p)\to A(p)\rceil\wedge\lceil\delta^*(p,\epsilon,p)\to A(p)\rceil\\
&=(0^\perp\vee(0\wedge A(q)))\wedge(\delta(p,\sigma,p)^\perp\vee(\delta(p,\sigma,p)\wedge A(p)))\wedge(1^\perp\vee(1\wedge A(p)))\\
&=1\wedge(v_{01}^\perp\vee(v_{01}\wedge v_{11}))\wedge A(p)\\
&=v_{01}^\perp\wedge v_{11}\\
&=v_{11}=A(p)
\end{aligned}$$

and

$$\begin{aligned}
&\wedge\{\lceil\delta^*(q,x,r)\to A(r)\rceil : x\in\Sigma^*, r\in Q\}\\
&=\bigwedge\{\lceil\delta^*(q,x,p)\to A(p)\rceil : x\in\Sigma^*\}\wedge\bigwedge\{\lceil\delta^*(q,x,q)\to A(q)\rceil : x\in\Sigma^*\}\\
&=\lceil 0\to A(p)\rceil\wedge\lceil\delta(q,\sigma,q)\to A(q)\rceil\wedge\lceil\delta^*(q,\epsilon,q)\to A(q)\rceil\\
&=(0^\perp\vee(0\wedge A(p)))\wedge(\delta(q,\sigma,q)^\perp\vee(\delta(q,\sigma,q)\wedge A(q)))\wedge(1^\perp\vee(1\wedge A(q)))\\
&=1\wedge(v_{10}^\perp\vee(v_{10}\wedge v_{00}))\wedge A(q)\\
&=v_{10}^\perp\wedge v_{00}\\
&=v_{00}=A(q).
\end{aligned}$$

That is,

$$A(p)=v_{11}=\wedge\{\lceil\delta^*(p,x,r)\to A(r)\rceil : x\in\Sigma^*, r\in Q\}$$

and

$$A(q)=v_{00}=\wedge\{\lceil\delta^*(q,x,r)\to A(r)\rceil : x\in\Sigma^*, r\in Q\}$$

which together, by means of Definition 2.3, implies that A is an l-valued subautomaton of $\mathcal{M}$.

In addition, in terms of Definition 2.2 and the definitions of δ and A above, we have

$$\begin{aligned}
S(A)(p) &= \bigvee\{\delta^*(q,x,p)\wedge A(q) : x\in\Sigma^*\}\vee\bigvee\{\delta^*(p,x,p)\wedge A(p) : x\in\Sigma^*\}\\
&=0\vee(\delta(p,\sigma,p)\wedge A(p))\vee(\delta(p,\epsilon,p)\wedge A(p))\\
&=(v_{01}\wedge v_{11})\vee(1\wedge v_{11})\\
&=v_{11}=A(p),
\end{aligned}\tag{29}$$

and similarly

$$S(A)(q)=v_{00}=A(q).\tag{30}$$

Therefore, we obtain

$$\begin{aligned}
S(S(A))(p) &= \bigvee\{\delta^*(p,x,p)\wedge S(A)(p) : x\in\Sigma^*\}\\
&=(\delta(p,\sigma,p)\wedge S(A)(p))\vee(\delta(p,\epsilon,p)\wedge S(A)(p))
\end{aligned}$$

$$= (v_{01} \wedge v_{11}) \vee (1 \wedge v_{11})$$
$$= v_{11} = S(A)(p), \tag{31}$$

and analogously

$$S(S(A))(q) = v_{00} = S(A)(q). \tag{32}$$

It follows from Eqs. (29)–(32) that

$$\overset{l}{\models} S(S(A)) \equiv S(A). \tag{33}$$

Remark 5.1. As Proposition 2.2 states, "*for any*" l-VFA $\mathcal{M} = (Q, \Sigma, \delta)$, and "*for any*" $A \in L^Q$, Eq. (33) holds if and only if L is a Boolean algebra. However, we have provided an example above that demonstrates that $\overset{l}{\models} S(S(A)) \equiv S(A)$ still holds for l_2-VFA $\mathcal{M}$ and l_2-valued subset A, with L being a nonBoolean algebra. Therefore, this shows that, though the truth-value lattice l_2 does *not* satisfy the distributivity, some fundamental properties in some l_2-VFAs may be still valid. This is a positive outcome in contrast to Proposition 2.2. Also, it implies that some propositions of classical computation theory are still preserved in the theory of computation based on a certain quantum logic (for example, with regard to the state space $\otimes^n \mathbf{C}^2$). Indeed, more examples can also be constructed for showing that those negative results may hold in some special cases. However, from the viewpoint of universality, it is still worthy of further consideration.

6. Concluding remarks

As a continuation of [36], this paper has dealt with reversibility in the setting of l-VFAs and established a basic framework of l-VPDAs. More specifically, (i) we have clarified the relationships between various reversibilities associated with quantum finite automata defined in the literature (although Bavel and Muller [11] considered analogous problem, the reversible finite automata defined by them were not required to be DFAs); (ii) we have elaborated reversibility of l-VFAs, and particularly, we have clarified the relationships among various retrievabilities in the setting of l-VFAs and showed that some of them are equivalent but for the others, they are equivalent if and only if the truth-value set satisfies a certain condition, which is an essential difference from classical situation, since all those retrievabilities are exactly equivalent in classical finite automata [5]; (iii) we have introduced l-VPDAs and showed that the class of the languages accepted by l-VPDAs by empty stack are the same as that accepted by l-VPDAs by final state; (iv) some examples of l-VFAs were presented, showing that those negative results we obtained may still hold in some l-VFAs .

The equivalence problem between probabilistic grammars and probabilistic automata were investigated by some authors (for example [40]). In the setting of quantum computing, Moore and Crutchfield [29] dealt with the equivalence relation between quantum automata and quantum grammars. Naturally, an issue worthy of further consideration is to deal with l-valued context-free grammars and their equivalence to l-VPDAs introduced in this paper. Furthermore, establishing systematically l-valued Turing machines is also a significant problem that should be considered in subsequent work.

Acknowledgements

The author would like to thank the Editor and the anonymous referees for their invaluable comments and suggestions that greatly helped to improve the quality of this paper. Also, I thank some friends who helped to improve the linguistic quality.

References

[1] D. Angluin, Inference of reversible languages, J. ACM 29 (1982) 741–765.
[2] A. Ambainis, R. Freivalds, One-way quantum finite automata: Strengths, weaknesses and generalizations, in: Proc. 39th Annu. Symp. Foundations of Computer Science, Palo Alfo, California, 1998, pp. 332–341. Also quant-ph/9802062, 1998.
[3] A. Ambainis, A. Nayak, A. Ta-Shma, U. Vazirani, Dense quantum coding and a lower bound for 1-way quantum automata, in: Proc. 31st Annu. ACM Symp. Theory of Computing, Atlanta, Georgia, 1999, pp. 376–383. Also quant-ph/9804043, 1998.

[4] A. Ambainis, J. Watrous, Two-way finite automata with quantum and classical states, Theoret. Comput. Sci. 287 (2002) 299–311.
[5] Z. Bavel, Introduction to the Theory of Automata, Reston Publishing Company, Inc., Reston, VA, 1983.
[6] H. Barnum, Quantum information processing and quantum logic: Towards mutual illumination. quant-ph/0203129.
[7] P. Benioff, The computer as a physical system: A microscopic quantum mechanical Hamiltonian model of computers as represented by Turing machines, J. Stat. Phys. 22 (1980) 563–591.
[8] C. Bennett, Logical reversibility of computation, IBM J. Res. Dev. 17 (1973) 525–532.
[9] A. Bertoni, M. Carpentieri, Analogies and differences between quantum and stochastic automata, Theoret. Comput. Sci. 262 (2001) 69–81.
[10] O. Brunet, P. Jorrand, Dynamic quantum logic for quantum programs, Internat. J. Quantum Inf. 2 (1) (2003) 45–54. Also quant-ph/0311143.
[11] Z. Bavel, D.E. Muller, Connectivity and reversibility in automata, J. ACM 17 (1970) 231–240.
[12] G. Birkhoff, J. von Neumann, The logic of quantum mechanics, Ann. of Math. 37 (1936) 823–843.
[13] A. Brodsky, N. Pippenger, Characterizations of 1-way quantum finite automata, SIAM J. Comput. 31 (2002) 1456–1478. Also quant-ph/9903014, 1999.
[14] E. Bernstein, U. Vazirani, Quantum complexity theory, SIAM J. Comput. 26 (1997) 1411–1473.
[15] G. Cattaneo, M.L. Dalla Chiara, R. Giuntini, R. Leporini, An unsharp logic from quantum computation. quant-ph/0201013.
[16] W. Cheng, J. Wang, Grammar theory based on quantum logic, Internat. J. Theoret. Phys. 42 (2003) 1677–1691.
[17] D. Deutsch, Quantum theory, the Church–Turing principle and the universal quantum computer, Proc. R. Soc. Lond. A 400 (1985) 97–117.
[18] M.L. Dalla Chiara, R. Giuntini, R. Leporini, Quantum computational logics: A survey. quant-ph/0305029.
[19] C.A. Ellis, Probabilistic languages and automata, Ph.D. Dissertation, University of Illinois, Urbana, IL, 1969.
[20] R.P. Feynman, Simulating physics with computers, Internat. J. Theoret. Phys. 21 (1982) 467–488.
[21] M. Golovkins, Quantum pushdown automata, in: Proc. 27th Conf. on Current Trends in Theory and Practice of Informatics, Milovy, in: Lecture Notes in Computer Science, vol. 1963, Springer-Verlag, Berlin, 2000, pp. 336–346.
[22] L. Grover, A fast quantum mechanical algorithms for datdbase search, in: Proc. 28th Annual ACM Symp. Theory of Computing, Philadelphia, PA, 1996, pp. 212–219.
[23] S. Gudder, Quantum computers, Internat. J. Theoret. Phys. 39 (2000) 2151–2177.
[24] J. Gruska, Quantum Computing, McGraw-Hill, London, 1999.
[25] J.E. Hopcroft, J.D. Ullman, Introduction to Automata Theory, Languages, and Computation, Addision-Wesley, New York, 1979.
[26] J. Karhumäki, Applications of finite automata, in: MFSC'02, in: Lecture Notes in Computer Science, vol. 2420, Springer, Berlin, 2002, pp. 40–58.
[27] A. Kondacs, J. Watrous, On the power of finite state automata, in: Proc. 38th IEEE Annu. Symp. Foundations of Computer Science, 1997, pp. 66–75.
[28] R.Q. Lu, H. Zheng, Lattices of quantum automata, Internat. J. Theoret. Phys. 42 (2003) 1425–1449.
[29] C. Moore, J.P. Crutchfield, Quantum automata and quantum grammars, Theoret. Comput. Sci. 237 (2000) 275–306. Also quant-ph/9707031, 1997.
[30] M.A. Nielsen, I.L. Chuang, Quantum Computation and Quantum Information, Cambridge University Press, Cambridge, 2000.
[31] H. Nishimura, T. Yamakami, An application of quantum finite automata to interactive proof system, in: Proc. 9th Internat. Conf. Implementation and Application of Automata, in: Lecture Notes in Computer Science, vol. 3317, Springer, Berlin, 2004, pp. 225–236. Also quant-ph/0410040, 2004.
[32] J. Pin, On reversible automata, in: Proc. Latin American Symp. Theoretical Informatics, in: Lecture Notes in Computer Science, vol. 583, Springer, Berlin, 1992, pp. 401–415.
[33] P. Pták, S. Pulmannová, Orthomodular Structures as Quantum Logics, Kluwer, Dordrecht, 1991.
[34] D.W. Qiu, Automata and grammars theory based on quantum logic, J. Softw. 14 (2003) 23–27.
[35] D.W. Qiu, Characterization of sequential quantum machines, Internat. J. Theoret. Phys. 41 (2002) 811–822.
[36] D.W. Qiu, Automata theory based on quantum logic: Some characterizations, Inform. and Comput. 190 (2004) 179–195.
[37] D.W. Qiu, M.S. Ying, Characterization of quantum automata, Theoret. Comput. Sci. 312 (2004) 479–489.
[38] M.O. Rabin, Probabilistic automata, Inf. Control 6 (1963) 230–244.
[39] L. Román, B. Rumbos, Quantum logic revisited, Found. Phys. 21 (1991) 727–734.
[40] A. Salomaa, Probabilistic and weighted grammars, Inf. Control 15 (1969) 529–544.
[41] E.S. Santos, Probabilistic grammars and automata, Inf. Control 21 (1972) 27–47.
[42] E. Santos, Relation of the Bell inequalities with quantum logic, hidden variable and information theory. quant-ph/0207062.
[43] P.W. Shor, Algorithm for quantum computation: Discrete logarithms and factoring, in: Proc. 37th IEEE Annu. Symp. on Foundations of Computer Science, 1994, pp. 124–134.
[44] A.C. Yao, Quantum circuit complexity, in: Proc. 34th IEEE Symp. Foundations of Computer Science, 1993, pp. 352–361.
[45] M.S. Ying, Automata theory based on quantum logic. (I), Internat. J. Theoret. Phys. 39 (2000) 981–991.
[46] M.S. Ying, Automata theory based on quantum logic. (II), Internat. J. Theoret. Phys. 39 (2000) 2545–2557.